SGS µProduct Support 11

Using the Z80 SIO in SDLC mode and in Bynary Synchronous Communications

SDLC MODE

Introduction

This μproduct support note describes the use of the Z80 SIO with the increasingly popular Synchronous Data Link Control (SDLC) communications protocol. A general description of the SDLC protocol and implementation of the protocol using the SIO are discussed. Desriptions for transmit and receive operations are given for use with simple control frame sequences.

The reader should be familiar with hardware aspects of the SIO such as interfacing to the CPU and modem.

Description

Data communication today requires a communication protocol that can transfer data quickly and reliably. One such protocol, Synchronous Data Link Control (SDLC), is the link control used by the IBM Systems Network Architecture (SNA) communication package. SDLC is actually a subset of the international Standards Organization (ISO) link control called High Level Data Link Control (HDLC), which is used for international data communication.

SDLC is a Bit-Oriented Protocol (BOP). It differs from Byte-Control Protocols (BCPs), such as bisync, in having a few bit patterns for control functions instead of several special character sequences. The attributes of the SDLC protocol are position dependent rather than character dependent, so control is determined by the location of the byte as well as by the bit pattern.

A character in SDLC is sent as an octet, a group of eight bits. Several octets combine to form a message frame in such a way that each octet belongs to a particular field. Each message frame consists of an opening flag, address, control, information, Frame Check Sequence (FCS), and closing flag fields. The flag field contains a unique binary pattern, 01111110, which indicates the beginning and end of a message frame. This pattern simplifies the hardware interface in receiving devices so that multiple devices connected to a common link do not conflict with one another. The receiving devices respond only after a valid flag character has been detected. Once communication is established for a particular device, the other devices ignore the message until the next flag character is detected.

The address field contains one or more octets that are used to select a particular station c the data link. An address of all 1s is a globa. address code that selects all the devices on the link. When a primary station sends a frame, the address field is used to select a secondary station. When a secondary station sends a message to the primary station, the address field contains the secondary station address, i.e., the source of the message.

The control field follows the address field and contains information about the type of frame being sent. The control field consists of one octet and is always present.

The information field consist of zero or more 8-bit octets and contains any actual data transferred. However, because of the limitations of the error-checking algorithm used in the frame-check sequence, maximum recommended block size is approximately 4096 octets.

The Frame Check Sequence (FCS) follows the information field or the control field, depending on the type of message frame se The FCS is a 16-bit Cyclic Redundancy Code (CRC) of the bits in the address, control, and information fields. The FCS is based on the CRC-CCITT code, which uses the polynomial $(X^{16} + X^{12} + X^5 + 1)$. The Z80 SIO contains the circuitry necessary to generate and check the FCS field.

Zero insertion/deletion is a feature of SDLC that allows any data pattern to be sent. Zero insertion occurs when five consecutive 1s in the data pattern are transmitted. After the fifth 1, a 0 is inserted before the next bit is sent. The data is not affected in any way except that

Description (Continued)

there is an extra 0 in the data stream.The receiver counts the 1s and deletes the 0 following the five consecutive 1s, thus restoring the original data pattern. Zero insertion and deletion is necessary because of the hardware constraint of searching for a flag character or abort sequence. Six 1s preceded and followed by a 0 indicate a flag character. Seven to 14 1s signify an abort, while an idle line (inactive) is indicated by 15 or more 1s. Under these three conditions, zero insertion/deletion is inhibited. Figure 2 illustrates the various line conditions.

LC protocol differs from other synchronous protocols with respect to frame timing. In bisync, for example, a host computer might interrupt transmission temporarily by sending sync characters instead of data. This suspended condition could continue as long as the receiver does not time out. With SDLC, however, it is illegal to send flags in the middle of a frame to idle the line. Such an occurence causes an error condition and disrupts orderly operation. Therefore, the transmitting device must send a complete frame without interruption. If a message cannot be completed, the primary station sends an abort and resumes message transmission later. These conditions are discussed later in the Programming section of this μproduct support note.

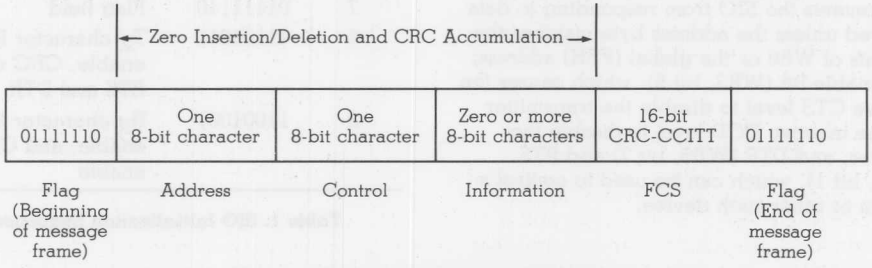
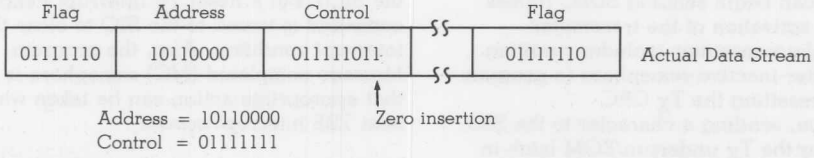


Figure 1. A Typical SDLC Message Frame Format



a) Zero insertion

XXXX11111110111110....
Abort Flag

b) Abort Condition

XXXX1111111111111111....
Idle

c) Idle Condition

Figure 2. Bit Patterns for Various Line Conditions

Programming the SIO

Implementation of the SDLC protocol with the Z80 SIO is simplified by the design of the SIO. This section discusses four areas of SIO programming: initialization, transmit operation, receive operation, and exception condition processing.

Initialization defines the basic mode of operation for the SIO. Table 1 shows the sequence of steps used to initialize the SIO, along with the necessary parameters. Since vectored interrupts are used, the SIO is programmed with the status affects vector (SAV) bit (WR1, bit 2) set.

Other function bits that can be included are the external interrupt enable bit (WR1, bit 0), which results in an interrupt for each DCD or CTS change, T_X underrun or abort change; address search bit (WR3, bit 2), which when set, prevents the SIO from responding to data received unless the address byte matches the contents of WR6 or the global (FFH) address; auto enable bit (WR3, bit 5), which causes the inactive CTS level to disable the transmitter and the inactive DCD level to disable the receiver; and DTR (WR5, bit 7) and RTS (WR5, bit 1), which can be used to control a modem or other such device.

Once the SIO is initialized and the transmitter is enabled, it sends flag characters continuously until a message begins transmission. These flag characters consist of the full 8-bit pattern. Although the SIO can receive flag characters with shared 0s (01111110111111101111110...), it can only transmit flag characters without shared 0s (01111110011111100111110...).

Register	Data	Function
0	00011000	Channel reset
2	(Vector)	Interrupt vector lower eight bits (channel B only)
4	00100000	SDLC mode
1	00011111	Interrupt control
6	(Address)	R_X address field
7	01111110	Flag field
5	11101011	T_X character length, enable, CRC enable RTS and DTR
3	11001001	R_X character length, enable, and CRC enable

Table 1. SIO Initialization Sequence

Transmit Operation

After the SIO has been initialized and enabled, it can begin sending SDLC frames by software activation of the transmitter. Activating the transmitter includes resetting the transmitter inactive semaphore (a program indicator), resetting the T_X CRC accumulation, sending a character to the SIO, and resetting the T_X underrun/EOM latch in the SIO. Figure 3 shows the sequence for transmitting a typical control message frame using interrupts.

When the SIO is loaded with the first data character (address byte), it stores the character in the T_X buffer until the current flag character has completed shifting. After the address byte is transferred into the shift register, a Transmit Buffer Empty (TBE) interrupt occurs. The program then loads the control character into the SIO and continues processing. The next TBE interrupt is ignored by the program (and no further data is sent to the SIO), but a Reset T_X Interrupt Pending command is issued to the SIO to clear the TBE interrupt condition. Also, the program Message completed (MC) semaphore is set so that appropriate action can be taken when the next TBE interrupt occurs.

When the last data character (the control byte) has been shifted out of the SIO, the T_X underrun/EOM latch is set because the SIO buffer was not loaded with a character on the previous TBE interrupt. As a result, an External/ Status Change (ESC) interrupt occurs and the SIO begins transmitting the FCS bytes automatically. In the ESC interrupt service routine, the program checks for other condition changes including CTS, DCD,and abort, and passes the status on to the program at the next-higher level.

Transmit Operation (Continued)

After the FCS bytes have been shifted out, the SIO generates a TBE interrupt to indicate that a flag character is being transmitted. The TBE interrupt service routine interprets the MC semaphore and determines that the frame has completed transmission. The program then clears the MC semaphore, sets the Transmitter inactive semaphore, starts a timer for a response from the receiving device, and clears the TBE interrupt condition. At this point, transmission of an SDLC message frame is complete and another message frame may be sent.

If the transmitter is to be turned off, the program must allow at least a two-character time delay before disabling the transmitter. This can be accomplished by connecting the SIO T_X clock line to the input of a counter and having the counter interrupt the CPU when the bit count expires.

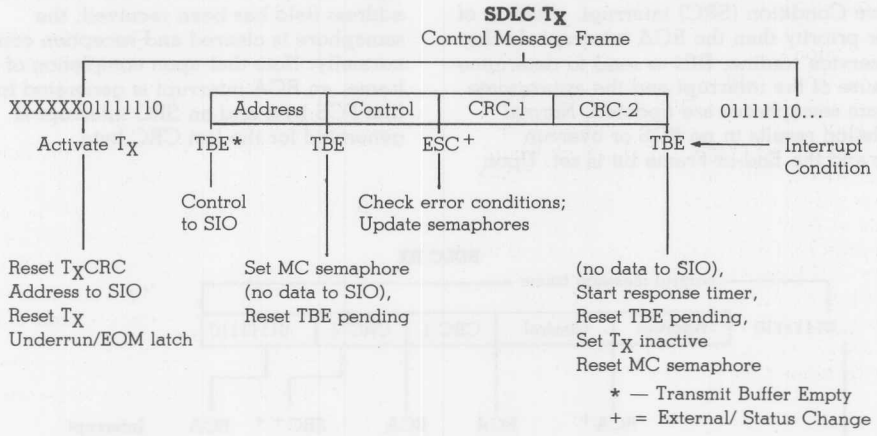


Figure 3. A Typical Transmit Control Frame Sequence

Receive Operation

The SDLC receive sequence is slightly less complex than the transmit sequence. To begin, the SIO enters Hunt mode when any of three conditions occurs: receive enable, abort detect, or a software command. In Hunt mode the SIO searches for flag characters, and when it detects a flag, the SIO generates an ESC interrupt. This interrupt can be used to signal line activation or the end of an abort condition, depending upon the previous receive condition. For example, when the SIO has been initialized, the receive circuitry is enabled and immediately begins searching for flag characters (Hunt mode operation). When the first flag is detected, the SIO exits from Hunt mode, which results in an ESC interrupt, and the SIO begins searching for the address field. If the SIO is programmed for Address Search mode and an address is received that does not match the programmed address byte in the SIO, the SIO does nothing until the next flag is found, after which the SIO again searches for an address match.

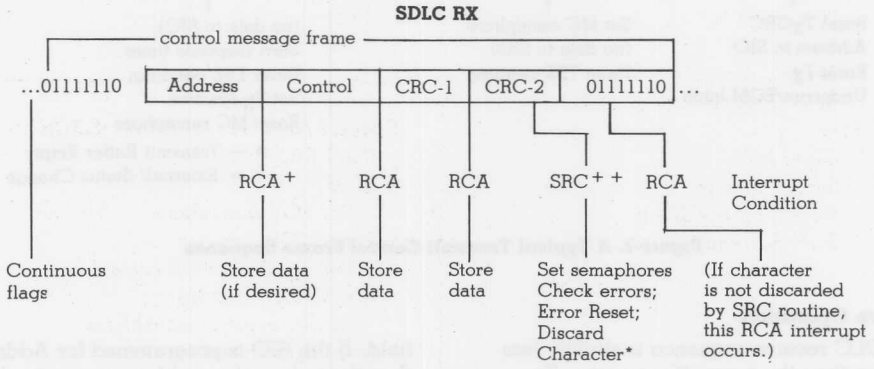
If the address field matches the address byte programmed into the SIO, the SIO generates a Receive Character Available (RCA) interrupt when the address byte is ready to be transferred from the SIO to the CPU. If the SIO is programmed to interrupt on all receive characters, it generates an RCA interrupt for each character received thereafter. It should be noted that the SIO generates the RCA interrupt when a character reaches the top of

Receive Operation (Continued)

the receive FIFO rather than when a character is transferred from the shift register to the FIFO. This means that if the FIFO is full of data, each character generates a separate RCA interrupt. This results in a more consistent software routine that does not need to check the receive FIFO, provided there is enough time between character transfers to allow the routine to complete the processing for each character.

After the last FCS byte of a frame is received and processed, the SIO generates a Special Receive Condition (SRC) interrupt, which is of higher priority than the RCA interrupt. In the SRC service routine, RR1 is read to determine the cause of the interrupt and the appropriate program semaphores are updated. Normal completion results in no FCS or overrun errors and the End-of-Frame bit is set. Upon completion of the SRC interrupt service routine, the program issues an Error Reset command to the SIO and reads the data port to discard the received data. If the data is not read and discarded, an RCA interrupt occurs. Now, a complete message frame and the first FCS byte are in the receive buffer.

Figure 4 shows the sequence for a typical control frame received by the SIO. If the address field byte is to be discarded, a program semaphore should initially be set to signal this to the RCA routine. After the address field has been received, the semaphore is cleared and reception continues normally. Note that upon completion of a frame, an RCA interrupt is generated for the first FCS byte and an SRC interrupt is generated for the last CRC byte.



NOTES

* The SRC routine normally reads the data character to clear the SIO buffer. This should be done after the program issues an Error Reset command.

\+ RCA = Receive Character Available

\+ + SRC = Special Receive Condition (higher priority than RCA)

Figure 4. A Typical Receive Control Frame Sequence

Receive Operation (Continued)

Table 2 lists the contents of the interrupt service routines used with the SIO. The wake routine is not an interrupt service routine but is a routine called by the program on the next higher level to begin frame transmission.

Once the wake routine is called, the program on the next higher level monitors the T_X active semaphore to determine when the current frame completes transmission and the next frame transmission can begin.

Wake:
- Clear T_X inactive semaphore
- Reset T_X CRC
- Data to SIO
  - (Address field byte)
- Reset T_X Underrun/EOM latch

External/Status Change (ESC):
- Clear DCD, CTS, abort semaphores
- If (abort)
  - Set abort semaphore
- Else if (DCD change)
  - Set DCD semaphore
- Else if (CTS change)
  - Set CTS semaphore

Transmit Buffer Empty (TBE):
- If (MC cleared)
- If (buffer not empty)
  - Data to SIO
- Else,
  - Set MC semaphore
  - Reset TBE condition
- Else,
  - Clear MC
  - Set T_X inactive
  - Reset TBE condition
  - Start Response timer

Receive Character Available (RCA):
- If (EOF)
  - Read and discard data
- Else,
  - Store data

Special Receive Condition (SRC):
- Read SIO RR1
- If (EOF)
  - Set EOF semaphore
- Else if (CRC error)
  - Set R_X CRC error semaphore
- Else if (R_X overrun)
  - Set R_X overrun semaphore
- Issue Error Reset
- Read data & discard

Table 2. SIO SDLC Interrupt Service Routines

Exception Condition Operation

Most of the exception conditions encountered in the SDLC protocol have been discussed in the previous sections. They include abort detect and DCD or CTS change. This section further describes some of the more unusual conditions.

DCD and CTS Change. The program handles DCD and CTS change by updating its semaphores each time an ESC interrupt occurs. In this manner, the program on the next higher level monitors the semaphores and determines a course of action based on what these semaphores indicate.

Abort and Idle Line Detect. Abort and idle line detect are a bit more complicated, since they result in similar interrupt operations. An abort occurs during a valid message frame. If the abort time is greater than 14 bits, an idle line is detected. This detection can be done by activating a timer when the ESC interrupt that signals a marking line occurs. If another ESC interrupt occurs before the timer times out, the line is in an abort condition. If the timer times out before another ESC interrupt occurs, then the line is idle and the program can pursue an appropriate course of action. A possible mechanism for implementing the timer function is to use a programmable counter that is tied to the receive clock line to count bits. The counter is programmed for eight clock transitions and is started as soon as the SIO interrupts the CPU with an abort condition. Only eight clock transitions need be counted because by the time the SIO generates the ESC interrupt, at least seven 1s have already passed. Figure 5 shows the abort/idle line timing and the interrupts resulting from the line changes.

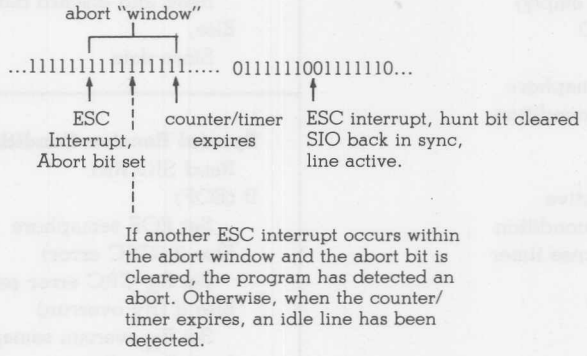


Figure 5. Abort/Idle Line Conditions

Conclusion

This μproduct support note describes implementation of the SDLC protocol using the SIO in an interrupt-driven enviroment. Descriptions for transmit and receive operations are given for use with simple control frame sequences. For frames that transfer data, the sequences are similar except for transmit, where a data character is sent to the SIO for a TBE interrupt. For receive, multiple RCA interrupts occur for each data byte received.

The Z80 SIO enhances system performance by minimizing CPU intervention during data transfers using the SDLC protocol. Performance can be improved further by using the Z80 DMA with the SIO, resulting in an efficient system configuration that reduces CPU interaction to a minimum.

program uses vectored interrupts to send a short SDLC control frame consisting of Address 9EH, Control 19H, and Data 81H. The response timer times the response of the receiving station after a message has been retransmits the message frame (if the retransmit count has not yet expired). This program transmits continuously until the processor is reset or interrupted by an external source.

```
                                                TEST SDLC
LOC   OBJ CODE M    STMT    SOURCE  STATEMENT
                       1    ;       SIO SDLC TEST PROGRAM
                       2
                       3    ; (0)   01-21-81/MDP                    INITIAL CREATION
                       4
                       5    ;       THIS PROGRAM SENDS ADDRESS 9EH, CONTROL 19H,
                       6    ;       AND DATA 81H CONTINUOUSLY USING THE Z80 VECTORED
                       7    ;       INTERRUPT MODE. THE SIO IS INITIALIZED TO USE
                       8    ;       SDLC WITH THE BAUD RATE CLOCK SUPPLIED BY
                       9    ;       HARDWARE INTERNAL TO THE SYSTEM.
                      10
                      11    ;       EQUATES
                      12
                      13    ADDRESS         EQU     9EH     ; ADDRESS FIELD
                      14    CTRL:   EQU     19H             ; CONTROL FIELD
                      14    DATA:   EQU     81H             ; INFORMATION FIELD
                      16    MSGLEN: EQU     1               ; MESSAGE LENGTH
                      17    RAM:    EQU     2000H           ; RAM ORIGIN
                      18    RAMSIZ: EQU     1000H           ; RAM SIZE
                      19    SIODA:  EQU     0               ; SIO PORT A DATA
                      20    SIOCA:  EQU     SIODA+1         ; SIO PORT A CTRL
                      21    SIODB:  EQU     SIODA+2         ; SIO PORT B DATA
                      22    SIOCB:  EQU     SIODB+1         ; SIO PORT B CTRL
                      23    CIOC:   EQU     8               ; CIO PORT C
                      24    CIOB:   EQU     CIOC+1          ; CIO PORT B
                      25    CIOA:   EQU     CIOC+2          ; CIO PORT A
                      26    CIOCTL: EQU     CIOC+3          ; CIO CTRL PORT
                      27    BAUD:   EQU     9600            ; ASYNC BAUD RATE
                      28    RATE:   EQU     BAUD/100
                      29    CIOCNT: EQU     9216/RATE
                      30    LITE:   EQU     0E0H            ; LIGHT PORT
                      31    RSPCNT: EQU     100             ; RESPONSE TIMER VALUE
                      32
                      33    ;       SIO PARAMETERS
                      34
                      35    SIOWR0: EQU     0
                      36            CHRES:  EQU     18H     ; CH.RESET CMD
                      37            ESCRES: EQU     10H     ; ESC RESET CMD
                      38            TBERES: EQU     28H     ; TBE RESET CMD
                      39            RETIA:  EQU     38H     ; RETI CH. A
                      40            ENINRX: EQU     20H     ; ENAB. INT. NEXT RX
                      41            SRCRES: EQU     30H     ; SRC RESET CMD
                      42            RCRCRE: EQU     40H     ; RX CRC RESET CMD
                      43            TCRCRE: EQU     80H     ; TX CRC RESET CMD
                      44            EOMRES: EQU     0C0H    ; EOM RESET CMD
                      45
                      46    SIOWR1: EQU     1
                      47            WREN:   EQU     80H     ; WAIT/RDY ENABLE
                      48            RDY:    EQU     40H     ; READY FUNCT.
                      49            WRONR:  EQU     20H     ; WAIT/RDY ON RX
                      50            RXIFC   EQU     8       ; RX INT. FIRST CHAR
```

```
                                TEST. SDLC
LOC   OBJ CODE M   STMT  SOURCE  STATEMENT
                     51          RXIAP:    EQU     10H     ; RX INT. ALL + PARITY
                     52          RXIA:     EQU     18H     ; RX INT. ALL
                     53          SIOSAV:   EQU     4       ; STATUS AFFECTS VECT.
                     54                                    ; (CH B ONLY)
                     55          TXI:      EQU     2       ; TX INT. ENABLE
                     56          EXTI:     EQU     1       ; EXT. INT. ENABLE
                     57
                     58    SIOWR2: EQU     2               ; (CH. B ONLY)
                     59
                     60    SIOWR3: EQU     3
                     61          RX8:      EQU     0C0H    ; RX 8 BITS
                     62          RX6:      EQU     80H     ; RX 6 BITS
                     63          RX7:      EQU     40H     ; RX 7 BITS
                     64          RX5:      EQU     0       ; RX 5 BITS
                     65          AUTOEN:   EQU     20H     ; AUTO ENABLES
                     66          HUNT:     EQU     10H     ; HUNT MODE
                     67          RXCRC:    EQU     8       ; RX CRC ENABLE
                     68          ADSRCH:   EQU     4       ; ADDR SEARCH
                     69          SYNINH:   EQU     2       ; SYNC LOAD INHIBIT
                     70          RXEN:     EQU     1       ; RX ENABLE
                     71
                     72    SIOWR4: EQU     4
                     73          X64:      EQU     0C0H    ; 64X CLOCK
                     74          X32:      EQU     80H     ; 32X CLOCK
                     75          X16:      EQU     40H     ; 16X CLOCK
                     76          X1:       EQU     0       ; 1X CLOCK
                     77          EXTSYN:   EQU     30H     ; EXT. SYNC ENABLE
                     78          SDLC:     EQU     20H     ; SDLC MODE
                     79          SYN16:    EQU     10H     ; 16 BIT SYNC
                     80          SYN8:     EQU     0       ; 8 BIT SYNC
                     81          STOP2:    EQU     0CH     ; 2 STOP BITS
                     82          STOP15:   EQU     8       ; 1.5 STOP BITS
                     83          STOP1:    EQU     4       ; 1 STOP BIT
                     84          SYNCEN:   EQU     0       ; SYNC ENABLE
                     85          EVEN:     EQU     2       ; EVEN PARITY
                     86          PARITY:   EQU     1       ; PARITY ENABLE
                     87
                     88    SIOWR5: EQU     5
                     89          DTR:      EQU     80H     ; ACTIVATE DTR
                     90          TX8:      EQU     60H     ; TX 8 BITS
                     91          TX6:      EQU     40H     ; TX 6 BITS
                     92          TX7:      EQU     20H     ; TX 7 BITS
                     93          TX5:      EQU     0       ; TX 5 BITS
                     94          BREAK:    EQU     10H     ; TX BREAK
                     95          TXEN:     EQU     8       ; TX ENABLE
                     96          CRC16:    EQU     4       ; CRC-16 MODE
                     97          RTS:      EQU     2       ; ACTIVATE RTS
                     98          TXCRC:    EQU     1       ; TX CRC ENABLE
                     99
                    100    SIOWR6: EQU     6               ; LOW SYNC OR ADDR
                    101
                    102    SIOWR7: EQU     7               ; HIGH SYNC OR FLAG
                    103
                    104    ;       SIOFLG = FLAGS FOR SIO STATUS
                    105
                    106    ;       BIT - - - SET CONDITION
                    107
                    108    ;       0         TX ACTIVE
                    109    ;       1         MESSAGE COMPLETE
                    110    ;       2         CTS ACTIVE
                    111    ;       3         DCD ACTIVE
```

```
                                                    TEST. SDLC
LOC   OBJ CODE M      STMT    SOURCE  STATEMENT
                       112    ;       4       ABORT DETECT
                       113    ;       5       RX OVERRUN ERROR
                       114    ;       6       RX CRC ERROR
                       115    ;       7       RX END OF FRAME
                       116
                       117
                       118    ;;              * * * MAIN PROGRAM * * *
                       119
0000                   120                    ORG     0
0000  C32000            121                    JP      BEGIN           ; GO MAIN PROGRAM
                       122
                       123    ;               INTERRUPT VECTORS
                       124    ;               (MUST START ON EVEN BOUNDARY)
                       125
0010                   126                    ORG     $ AND. 0FFF0H. OR 10H
                       127    INTVEC:
                       128    SIOVEC:
0010  9C00             129                    DEFW    CHBTBE
0012  D100             130                    DEFW    CHBESC
0014  0101             131                    DEFW    CHBRCA
0016  0F01             132                    DEFW    CHBSRC
0018  3B01             133                    DEFW    CHATBE
001A  4301             134                    DEFW    CHAESC
001C  4B01             135                    DEFW    CHARCA
001E  5101             136                    DEFW    CHASRC
                       137
                       138    BEGIN:
0020  314020           139                    LD      SP, STAK        ; INIT SP
0023  ED5E             140                    IM      2               ; VECTOR INTERRUPT MODE
0025  3E00             141                    LD      A, INTVEC/256   : UPPER VECTOR BYTE
0027  ED47             142                    LD      I, A
0029  214520           143                    LD      HL.BUFFER
002C  369E             144                    LD      (HL), ADDRESS   ; STORE ADDRESS
002E  23               145                    INC     HL
002F  3619             146                    LD      (HL), CTRL      ; STORE CTRL BYTE
0031  23               147                    INC     HL
0032  3681             148                    LD      (HL), DATA      ; STORE DATA BYTE
0034  CD4C00           149                    CALL    INIT            ; INIT DEVICES
0037  218720           150                    LD      HL, RBUF        ; SETUP READ BUFFER
003A  228520           151                    LD      (RBPTR), HL
                       152    LOOP:
003D  213D00           153                    LD      HL, LOOP        ; SETUP STACK FOR RETURN
0040  E5               154                    PUSH    HL
0041  CD7D00           155                    CALL    WAKE            ; WAKE TX
                       156    LOOP1:
      3A4020           157                    LD      A, (SIOFLG)     ; CHECK TX ACTIVE FLAG
0047  CB47             158                    BIT     0, A
0049  20F9             159                    JR      NZ, LOOP1       ; LOOP IF TX ACTIVE
004B  C9               160                    RET
                       161
                       162    INIT:
                       163    SIOINI:
004C  217001           164                    LD      HL, SIOTA       ; INIT CH. A
004F  0E01             165                    LD      C, SIOCA
0051  060A             166                    LD      B, SIOEA-SIOTA
0053  EDB3             167                    OTIR
0055  217A01           168                    LD      HL, SIOTB       ; INIT CH. B
0058  0E03             169                    LD      C, SIOCB
005A  0610             170                    LD      B, SIOEB-SIOTB
005C  EDB3             171                    OTIR
005E  3E00             172                    LD      A, 0            ; CLEAR FLAG BYTE
```

```
                                TEST. SDLC
LOC   OBJ CODE M    STMT  SOURCE STATEMENT
0060  324020        173           LD      (SIOFLG), A
                    174   CIOINI:
0063  DB0B          175           IN      A, (CIOCTL)     ; INSURE STATE 0
0065  AF            176           XOR     A               ;POINT TO REG 0
0066  D30B          177           OUT     (CIOCTL), A
0068  DB0B          178           IN      A, (CIOCTL)     ; CLEAR RESET OR STATE 0
006A  AF            179   XOR     A
006B  D30B          180           OUT     (CIOCTL), A     ; POINT TO REG 0
006D  3C            181           INC     A               ; WRITE RESET
006E  D30B          182           OUT     (CIOCTL), A
0070  AF            183           XOR     A               ;CLEAR RESET COND.
0071  D30B          184           OUT     (CIOCTL), A
0073  218A01        185           LD      HL, CLST        ; INIT CIO
0076  060E          186           LD      B, CEND-CLST
0078  0E0B          187           LD      C, CIOCTL
007A  EDB3          188           OTIR
007C  C9            189           RET
                    190
                    191   WAKE:
007D  3A4020        192           LD      A, (SIOFLG)     ; SET ACTIVE FLAG
0080  CBC7          193           SET     0, A
0082  324020        194           LD      (SIOFLG), A
0085  214520        195           LD      HL, BUFFER      ; SET BUFFER PTR
0088  224320        196           LD      (BUFPTR), HL
008B  3E03          197           LD      A, 2+MSGLEN     ; SET BYTE COUNT
008D  324120        198           LD      (BYTES), A
0090  3E80          199           LD      A, TCRCRE       ; CLEAR TX CRC
0092  D303          200           OUT     (SIOCB) A
0094  CD9C00        201           CALL    CHBTBE          ; START TRANSMIT
0097  3EC0          202           LD      A, EOMRES       ; RESET EOM LATCH
0099  D303          203           OUT     (SIOCB), A
009B  C9            204           RET
                    205
                    206
                    207   ; ;     INTERRUPT SERVICE ROUTINES
                    208
                    209   CHBTBE:
009C  CD5901        210           CALL    SAVE            ; CH. B TX BUFFER EMPTY
009F  214020        211           LD      HL, SIOFLG      ; POINT TO FLAG BYTE
00A2  CB4E          212           BIT     1, (HL)         ; CHECK MC FLAG
00A4  201D          213           JR      NZ, CHBTB2      ; BRANCH IF MESSAGE COMPLETE
00A6  3A4120        214           LD      A, (BYTES)      ; CHECK BYTE COUNT
00A9  B7            215           OR      A
00AA  280F          216           JR      Z, CHBTB1       ; BRANCH IF DATA DONE
00AC  3D            217           DEC     A
00AD  324120        218           LD      (BYTES), A
00B0  2A4320        219           LD      HL, (BUFPTR)
00B3  7E            220           LD      A, (HL)
00B4  D302          221           OUT     (SIODB), A
00B6  23            222           INC     HL
00B7  224320        223           LD      (BUFPTR), HL
00BA  C9            224           RET
                    225   CHBTB1:
00BB  CBCE          226           SET     1, (HL)         ; SET MC FLAG
00BD  3EC0          227           LD      A, EOMRES
00BF  D303          228           OUT     (SIOCB), A
00C1  1809          229           JR      CHBTB3
                    230   CHBTB2:
00C3  CB8E          231           RES     1, (HL)         ; CLEAR MC FLAG
00C5  CB86          232           RES     0, (HL)         ; SET TX INACTIVE
00C7  3E64          233           LD      A. RSPCNT       ; START RESPONSE TIMER
00C9  324220        234           LD      (RSPTMR), A
```

```
                                                    TEST. SDLC
LOC     OBJ CODE M      STMT    SOURCE  STATEMENT
                        235     CHBTB3:
00CC    3E28            236             LD      A, TBERES       ; RESET TBE INT. PEND.
00CE    D303            237             OUT     (SIOCB), A
00D0    C9              238             RET
                        239
                        240     CHBESC:
00D1    CD5901          241             CALL    SAVE            ; CH. B EXTERNAL/STATUS CHG
                                                                    CHG
00D4    214020          242             LD      HL, SIOFLG      ; GET FLAG BYTE
00D7    CB96            243             RES     2, (HL)
00D9    CB9E            244             RES     3, (HL)
00DB    CBA6            245             RES     4, (HL)
00DD    DB03            246             IN      A, (SIOCB)      ; READ RR0
00DF    47              247             LD      B, A            ; STORE IN %B
        CB58            248             BIT     3, B            ; CHECK DCD BIT
        C4FB00          249             CALL    NZ, SETDCD
00E5    CB68            250             BIT     5, B            ; CHECK CTS BIT
00E7    C4FE00          251             CALL    NZ, SETCTS
00EA    CB78            252             BIT     7, B            ; CHECK ABORT BIT
00EC    C4FB00          253             CALL    NZ, SETABT
00EF    CB4E            254             BIT     1, (HL)         ; CHECK MC FLAG
00F1    2800            255             JR      Z, CHBES1       ; BRANCH IF CLEAR
                        256     CHBES1:
00F3    3E10            257             LD      A, ESCRES       ; RESET ESC
00F5    D303            258             OUT     (SIOCB), A
00F7    C9              259             RET
                        260     SETABT:
00F8    CBE6            261             SET     4, (HL)
00FA    C9              262             RET
                        263     SETDCD:
00FB    CBDE            264             SET     3, (HL)
00FD    C9              265             RET
                        266     SETCTS:
00FE    CBD6            267             SET     2, (HL)
0100    C9              268             RET
                        269
                        270     CHBRCA:
0101    CD5901          271             CALL    SAVE            ; CH. B RX CHAR AVAIL.
0104    DB02            272             IN      A, (SIODB)
0106    2A8520          273             LD      HL, (RBPTR)     ; GET READ BUFF PTR
0109    77              274             LD      (HL), A
010A    23              275             INC     HL
010B    228520          276             LD      (RBPTR), HL
010E    C9              277             RET
                        278
                        279     CHBSRC:
010F    CD5901          280             CALL    SAVE            ; CH. B SPECIAL RX COND.
0112    3E01            281             LD      A, 1
0114    D303            282             OUT     (SIOCB), A      ; READ RR1
0116    DB03            283             IN      A, (SIOCB)
0118    47              284             LD      B, A            ; SAVE IN %B
0119    214020          285             LD      HL, SIOFLG
011C    CBB6            286             RES     6, (HL)         ; CLEAR CRC ERROR FLAG
011E    CB78            287             BIT     7, B            ; CHECK EOF BIT
0120    C43801          288             CALL    NZ, SETEFF      ; BRANCH IF NOT EOF
0123    CB70            289             BIT     6, B            ; CHECK CRC ERROR
0125    C43501          290             CALL    NZ, SETCRC
0128    CB68            291             BIT     5, B            ; CHECK OVERRUN BIT
012A    C43201          292             CALL    NZ, SETOVR
                        293     CHBSR1:
012D    3E30            294             LD      A, SRCRES       ; ERROR RESET CMD
```

```
                                        TEST. SDLC
LOC   OBJ CODE M    STMT    SOURCE      STATEMENT
012F  D303           295                OUT       (SIOCB), A
0131  C9             296                RET
                     297    SETOVR:
0132  CBEE           298                SET       5, (HL)
0134  C9             299                RET
                     300    SETCRC:
0135  CBF6           301                SET       6, (HL)
0137  C9             302                RET
                     303    SETEFF:
0138  CBFE           304                SET       7, (HL)
013A  C9             305                RET
                     306
                     307    CHATBE:
013B  CD5901         308                CALL      SAVE          ; CH. A TX BUFFER EMPTY
013E  3E28           309                LD        A, TBERES
0140  D301           310                OUT       (SIOCA), A
0142  C9             311                RET
                     312
                     313    CHAESC:
0143  CD5901         314                CALL      SAVE          ; CH. A EXTERNAL/STATUS CHG
0146  3E10           315                LD        A, ESCRES
0148  D301           316                OUT       (SIOCA), A
014A  C9             317                RET
                     318
                     319    CHARCA:
014B  CD5901         320                CALL      SAVE          ; CH. A RX CHAR AVAIL.
014E  DB00           321                IN        A, (SIODA)
0150  C9             322                RET
                     323
                     324    CHASRC:
0151  CD5901         325                CALL      SAVE          ; CH. B SPECIAL RX COND.
0154  3E30           326                LD        A, SRCRES
0156  D301           327                OUT       (SIOCA), A
0158  C9             328                RET
                     329
                     330    ;           SAVE REGISTER ROUTINE
                     331
                     332    SAVE:
0159  E3             333                EX        (SP), HL      ; SP = HL
015A  D5             334                PHUS      DE            ;      DE
015B  C5             335                PUSH      BC            ;      BC
015C  F5             336                PUSH      AF            ;      AF
015D  DDE5           337                PUSH      IX            ;      IX
015F  FDE5           338                PUSH      IY            ;      IY
0161  CD6F01         339                CALL      GO            ;      PC
0164  FDE1           340                POP       IY
0166  DDE1           341                POP       IX
0168  F1             342                POP       AF
0169  C1             343                POP       BC
016A  D1             344                POP       DE
016B  E1             345                POP       HL
016C  FB             346                EI
016D  ED4D           347                RETI
                     348
                     349    GO:
016F  E9             350                JP        (HL)
                     351
                     352
                     353    ; ;         CONSTANTS
                     354
                     355    SIOTA:
0170  00             356                DEFB      SIOWR0        ; CHAN. RESET
```

```
LOC  OBJ CODE M  STMT  SOURCE  STATEMENT
0171 18          357           DEFB    CHRES
0172 01          358           DEFB    SIOWR1          ; CHAN. CHARACS.
0173 D2          359           DEFB    WREN+RDY+RXIAP+TXI
0174 04          360           DEFB    SIOWR4          ; MODE
0175 4F          361           DEFB    X16+STOP2+EVEN+PARITY
0176 05          362           DEFB    SIOWR5          ; TX PARAMS.
0177 AA          363           DEFB    DTR+TX7+TXEN+RTS
0178 03          364           DEFB    SIOWR3          ; RX PARAMS.
0179 41          365           DEFB    RX7+RXEN
                 366   SIOEA:  EQU     $
                 367
                 368   SIOTB:
017A 00          369           DEFB    SIOWR0          ; CHAN. RESET
017B 18          370           DEFB    CHRES
017C 02          371           DEFB    SIOWR2          ; VECTOR REG.
017D 10          372           DEFB    SIOVEC. AND 255
017E 04          373           DEFB    SIOWR4          ; MODE
017F 20          374           DEFB    X1+SDLC+SYNCEN
0180 01          375           DEFB    SIOWR1          ; CHAN. CHARACS.
0181 1F          376           DEFB    RXIA+SIOSAV+TXI+EXTI
0182 06          377           DEFB    SIOWR6          ; ADDRESS
0183 9E          378           DEFB    ADDRESS
0184 07          379           DEFB    SIOWR7          ; FLAG
0185 7E          380           DEFB    01111110B
0186 05          381           DEFB    SIOWR5          ; TX PARAMS.
0187 EB          382           DEFB    DTR+TX8+TXEN+RTS+TXCRC
0188 03          383           DEFB    SIOWR3          ; RX PARAMS.
0189 C1          384           DEFB    RX8+RXEN
                 385   SIOEB:  EQU     $
                 386
                 387   CLST:
018A 28          388           DEFB    28H             ; PORT B MODE
018B 00          389           DEFB    00000000B
018C 2B          390           DEFB    2BH             ; DATA DIRECTION
018D EE          391           DEFB    11101110B
018E 1C          392           DEFB    1CH             ; CT1 MODE
018F C2          393           DEFB    11000010B
0190 16          394           DEFB    16H             ; CT1 TC MSB
0191 00          395           DEFB    0
0192 17          396           DEFB    17H             ;       LSB
0193 60          397           DEFB    CIOCNT
0194 01          398           DEFB    1               ; MASTER CONFIG. REG.
0195 F0          399           DEFB    11110000B
0196 0A          400           DEFB    10              ; CT1 TRIGGER
0197 06          401           DEFB    00000110B
                 402   CEND:   EQU     $
                 403
                 404
                 405   ;;      DATA AREA
                 406
2000             407           ORG     RAM
2000             408           DEFS    64              ; STACK AREA
                 409   STAK:   EQU     $
2040             410   SIOFLG: DEFS    1               ; SIO FLAG BYTE
2041             411   BYTES:  DEFS    1               ; BUFFER BYTE COUNT
2042             412   RSPTMR: DEFS    1               ; RESPONSE TIMER
2043             413   BUFPTR: DEFS    2               ; BUFFER POINTER
2045             414   BUFFER: DEFS    64              ; BUFFER
2085             415   RBPTR:  DEFS    2               ; READ BUFF PTR
                 416   RBUF:   EQU     $
                 417
                 418           END
```

Binary Synchronous

A popular communication protocol used to exchange information between data processing devices has been in use for some time. This protocol, developed by IBM, is called binary synchronous protocol, or bisync. The Z80 SIO provides a flexible and powerful tool for the implementation of the bisync protocol. However, there are some design considerations that require special attention. This paper will discuss these design considerations and offer an approach to using bisync with the Z80 SIO.

Bisync is a character-oriented protocol with information transmitted in blocks between two (or more) data communication devices. The medium through which this information is conveyed is called the data link. The particular data link discussed in this paper is a point-to-point link using the ASCII transmission code. Other codes, such as EBCDIC, are not covered, but the format for bisync is basically the same. The data link consists of a master station (usually a computer) and a slave station (usually a terminal) with the associated communication gear in between modems, phone lines, etc. The master station controls message flow by polling and selecting the slave station. Polling involves sending a general request message to the slave station(s) to determine whether or not any of the slaves have data to send (traffic). If a slave station does have traffic, it responds to the poll and the master can then select that particular slave for information exchange. Slaves can only respond to a master device and cannot initiate communication on the data link.

Information is exchanged by means of a well-defined block structure. Message blocks consist of a header, body, and trailer (Figure 6). The header is made of two or more SYN characters (hence the name bisync), a start of header (SOH) character, and addressing and control information for a particular slave station.

The body begins with a start of text (STX) character and encompasses the entire text information. The body generally contains ASCII text data, although 8-bit binary data can be transmitted using transparent text mode.

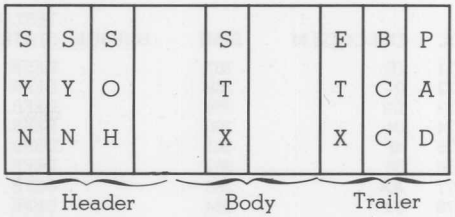


Figure 6. Basic Message Block Format for Bisync Protocol

The trailer contains the end of text (ETX) character and the block check character (BCC). The BCC is used for detecting errors through "cyclic redundancy checking" (CR or "longitudal redundancy checking" (LRC).

Error detection is essential when transferring information between data processing equipment. Since ASCII specifies only seven bits for its code, the eighth bit is used for vertical redundancy checking (VRC), more commonly known as character parity. In synchronous communications, character parity is generally odd, whereas in asynchronous communications it is even. Figure 7 shows typical ASCII characters with parity. The SIO can be programmed for 7-bit characters with odd parity enabled to minimize software overhead.

Because VRC applies only to the individual character, the entire message block has an LRC that makes up the BCC. The LRC is a simple bit position checksum where the number of 1s for each position (0 through 6) is even for a block of data. Since the BCC is a character, LRC is subject to the same character parity rules as the rest of the data block. The LRC includes all characters,

0	1	1	0	0	1	0	0	1	0	1	1	0	1	0	1
LSB						MSB	PARITY	LSB						MSB	PARITY

Figure 7. Odd VRC
Number of 1s should be odd.

except SYN, starting with the first character after SOH or STX and up to and including EXT in the trailer (Figure 8). Since the SIO cannot calculate the LRC, the task is left up to the user. LRC can be generated on a microprocessor with little effort by taking the message block and XORing the data with an initial value of zero to provide even LRC.

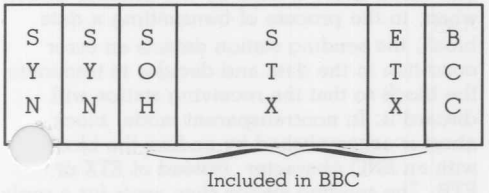


Figure 8. Characters Included in BBC

Another type of BBC is generated by a cyclic redundancy check (CRC), which results in a more powerful method of block checking. CRC-12 is used for 6-bit transmission code and CRC-16 is used for 8-bit transmission code. CRC is used in lieu of character parity and LRC, as with transparent text mode operation.

The remainder of this paper illustrates how to use the SIO in three special cases of the bisync protocol: transparent text mode, abort/interrupt procedures, and error recovery procedures.

Transparent text mode is useful in bisync when information exchanged between master and slave is not ASCII data. For example, a binary data file (object program) might be sent from master to slave. ASCII transmission c is only seven bits long making it difficult to send 8-bit binary data. One alternative is to convert the binary data to ASCII hex format at the master, transmit it to the slave and reconvert it back into binary at the slave.However, two disadvantages result from this. First,the master and slave require a means of conversion, by either software or hardware, adding cost to the data link. Since the slave (terminal) is burdened most by this, such an approach is usually not feasible. The other disadvantage is that the exchange of information is slower since two (or more) ASCII characters are sent for every eight bits of binary data. The bisync protocol has provisions for sending 8-bit binary data by using transparent text mode transmission. In this mode, character parity is disabled, allowing the full eight bits to be used for data. However, to allow control within the constraints of the protocol, there are certain limitations on the binary data pattern. The primary difference is that during transparent mode some communication control characters are preceded by a DLE character, actually making the control characters a two-character sequence. To distinguish a data byte from a control DLE, the protocol specifies insertion of another DLE. The receiver then throws away the first DLE, keeping the second as data. Table 3 shows the communication control characters that are valid during transparent mode.

Another character change occurs when the SYN character is used for line fill. Normally, the SYN character is ignored, but during transparent mode the SYN is preceded by a DLE, and both are consequently ignored by the receiver. In the event that the CPU does not have a character ready to send, the SIO automatically inserts SYN characters into the data stream. With the SIO programmed for 16-bit sync characters, two syncs are sent from the SIO (write registers WR6 and WR7) when its transmit buffer is empty. In transparent mode, the user must change WR6 and WR7 to DLE, SYN in order for the SIO to provide the proper line fill characters. In accordance with the ANSI standard, line fill characters are not included in the SIO CRC calculation during transmit. During reception in transparent mode, the software must disable CRC accumulation when the DLE SYN character sequence is detected.

DLE	STX	Start of transparent text
DLE	ETB	End of transparent text block
DLE	ETX	End of transparent text
DLE	SYN	Idle sync
DLE	ENQ	Enquiry
DLE	DLE	DLE data
DLE	SOH	Start of transparent header

Table 3. Control Codes Used in Transparent Mode

Binary Synchronous (Continued)

While in transparent mode, the user must be concerned with the error detection codes. If parity is enabled in the SIO normally, it must be disabled during transparent mode. This change in SIO operation affects both transmit and receive and should therefore be considered if using full duplex.

Since the SIO allows CRC enable/disable on the fly, the software can easily control CRC accumulation in both receive and transmit. During transmit, the CRC must be enabled/disabled before the character is transferred into the serial shift register. During receive, the CRC accumulation is delayed eight bits. After the character is transferred from the serial shift register into the buffer, the user has to read that character, decide whether or not to continue CRC accumulation, and disable/enable CRC before the next character is transferred to the buffer. This is not generally a problem, since character transfers occur about every 833 microseconds at 9600 baud. Table 4 shows the characters included and omitted in the CRC during transparent mode.

Omitted from CRC		Included in CRC
DLE	SYN	DLE of DLE DLE
DLE	SOH	ETX of DLE ETX
DLE	STX*	ETB of DLE ETB
		STX of DLE STX**

* If not preceded by transparent header within same block

** If preceded by DLE SOH within same block

Table 2. Character Included/Omitted in CRC During Transparent Mode

When CRC accumulation is to be resumed, the software should enable CRC before the desired character is transferred to the receive buffer. For example, suppose a DLE pair is received during transparent text mode. The SIO generates an interrupt when the first DLE is transferred to the receive buffer. The driver program reads the DLE and immediately disables CRC. When the next interrupt occurs, the driver reads the second DLE and immediately enables CRC to include the second DLE into the CRC accumulation

The second category of interest includes abort and interrupt procedures. There are two types of aborts: block abort and sending station abort. There are three types of interrupts: termination interrupt, reverse interrupt and temporary interrupt.

The block abort is used by the sending station when, in the process of transmitting a data block, the sending station detects an error condition in the data and decides to terminate the block so that the receiving station will discard it. In nontransparent mode, block abort is accomplished by ending the block with an ENQ character, instead of ETX or ETB. The sending station then waits for a reply from the receiver, which should be a NAK. The transparent mode procedure is identical except that a DLE ENQ character sequence is used. Since a block abort puts the data link back in nontransparent mode, NAK is the valid response the receiver should send in both transparent and nontransparent modes.

The sending station abort is similar to the block abort, except that the sending station does not necessarily do a block abort but simply ends the current message block, waits for as response or timeout, and then sends an EOT to regain control of the data link. The sending station abort is useful when transmission to a particular receiver is necessary due to a higher priority message, buffer overflow condition, error detection, etc. Once the sending station abort sequence is made, the master can perform any data link control function.

From the receiver side, a termination interrupt causes the sending station to stop transmission. Such a procedure is useful when the receiver cannot accept any more data or incurs an error condition, such as paper jam, card jam, hardware error, etc. To accomplish a termination interrupt, the receiving station sends an EOT instead of the normal response. The EOT resets all stations on the link and allows the master to issue any control sequence.

The reverse interrupt (RINT) is used when the receiving station needs to transmit during reception of several message blocks. The RINT

a DLE "<" character sequence to signal an affirmative acknowledgement and to stop transmission of data. Some exceptions and a more detailed description of RINT can be found in the ANSI standard.

The temporary interrupt procedure, WACK (Wait Before Sending Positive Acknowledge), is used by the receiving station to indicate positive acknowledgement and an inability to receive more data. Such a response may be necessary when the receiving station cannot accept data continuously, such as during a printing operation. The WACK consists of a DLE ";" character sequence and is sent in place of an ACK or ACKn. The sending station then sends ENQs (Enquiry) until the receiving station stops sending WACKs. The sending station can resume transmitting data when the receiving station sends an ACK or ACKn.

Recovery procedures provide a means of preventing data link instability. The recovery mechanism consists mainly of timers, grouped into four basic areas, and a NAK counter. The NAK counter is used to prevent repeated NAKs from inhibiting further communications. The sending unit counts how many NAKs it receives for a particular data block so that after a predetermined number of retries, it can recover and pursue another course of action. The particular count value and course of action taken when the count expires are left up to the user.

Four timers (timer A or response timer, timer B or receiver timer, timer C or gross timer, and timer D or no activity timer) prevent the data link from getting "hung" or going idle for extended periods of time. Generally, the shortest interval is used with timer A, and the longest interval is used with timer D. For maximum system efficiency, however, the receiver timer (timer B) should timeout before the response timer (timer A). The particular implementation of these timers varies from system to system, and some flexibility of exact timer values is left up to the user.

Since it is assumed that interrupts will be used reinitialized each time a character is received (receive interrupt). The same applies for the response timer, except that when a timeout occurs, the transmit driver has several options to follow.

If the SIO is set to transmit CRC on transmit underrun, then the driver could simply set its flags and not fill the buffer. This allows a normal exit, since the SIO will then send its CRC bytes. If the SIO is set to not transmit CRC on transmit underrun, then it sends sync characters (SYN SYN or DLE SYN, whichever was last written to WR6 and WR7) until the transmit buffer is filled or transmit data is set to marking.

In any event, enough time must be allowed after CRC is sent so that the receiver can properly decode CRC. Because of the character delay in the SIO during CRC accumulation, about 20 clock cycles are necessary after the last CRC byte is sent to ensure adequate decoding time.The SIO could be programmed to send pad characters either by disabling parity and sending 8-bit FFs (hex) or by filling WR6 and WR7 with FF hex. If enabled, the SIO automatically sends whatever is in its sync registers upon transmit underrun. Multiple message blocks do not have to be separated by pad characters as long as CRC is valid for the previous message block. However, to insure adeguate time for the receiver to process CRC, it is recommended that at least two pad characters follow the last character of a block.

Using the SIO for the bisync protocol is fairly straightforward. Care should be exercised when using the SIO in transparent text mode, but the implementation is greatly simplified by the SIO's flexibility, as compared to other serial communications ICs. The CRC capabilities of the SIO provide a powerful means of maintaining maximum data integrity with minimum software overhead. Coupled with the DMA and the interrupt capabilities of the Z80® processor, the user will find the SIO an excellent choice in serving data communication needs.

SGS-ATES GROUP OF COMPANIES
Italy - France - Germany - Malta - Malaysia - Singapore - Sweden - Switzerland - United Kingdom - U.S.A.